UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/730,975	12/10/2003	Efstratios Tsantilis	11884/406401	5120

53000 7590 06/11/2008

KENYON & KENYON LLP
1500 K STREET N.W.
WASHINGTON, DC 20005

EXAMINER	
WANG, JUE S	

ART UNIT	PAPER NUMBER
2193	

MAIL DATE	DELIVERY MODE
06/11/2008	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/730,975	TSANTILIS, EFSTRATIOS	
	Examiner	Art Unit	
	JUE S. WANG	2193	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *14 March 2008*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-35* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-35* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *10 December 2003* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-35 have been examined.

Claim Rejections - 35 USC § 112

2. The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

3. Claims 1-13, 25-27, 29-31, and 32-35 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

A. The following lacks antecedent basis in the claims:

i. Claim 1, "the updated version" in line 10.

ii. Claim 2, "the updated version" in line 2.

iii. Claim 29, "the machine-readable medium" in line 2.

iv. Claim 32, "the software environment" in line 17.

B. The following claim language is indefinite and not clear:

i. As per claim 1, lines 7-8, the phrase "comparing successively downloaded versions of the snapshots to detect at least one difference between them" is used. This limitation is not clearly understood because it is not clear how the comparison is performed when there are more than two snapshots (i.e., is every one of the downloaded snapshots compared to every other downloaded snapshots, or is every one of the downloaded snapshots compared to every other downloaded snapshots with a later version number than its own version?).

ii. As per claim 7, line 5, claim 12, line 5, claim 25, line 5, the phrase "comparing successively downloaded versions of the snapshots; detecting at least one difference between the snapshots" is used. This limitation is not clearly understood because it is not clear how the comparison is performed when there are more than two snapshots (i.e., is every one of the downloaded snapshots compared to every other downloaded snapshots, or is every one of the downloaded snapshots compared to every other downloaded snapshots with a later version number than its own version?).

iii. As per claim 7, lines 7-8, the phrase "determining an overall backward compatibility score of successive versions based on the detected difference ratings" is used. This limitation is not clearly understood because it is not clear what a successive version means (i.e., a snapshot that is made at a later time or is downloaded at a later time?) and it is not clear if one overall backward compatibility score is determined for all successive versions or one overall backward compatibility score is determined for every successive version.

iv. As per claim 12, line 7, the phrase "successive version of the object interface" is used. This limitation is not clearly understood because it is not clear what a successive version means (i.e., a snapshot that is made at a later time or is downloaded at a later time?).

v. As per claim 25, line 8, the phrase "successive snapshot" is used. This limitation is not clearly understood because it is not clear what a successive

snapshot means (i.e., a snapshot that is made at a later time or is downloaded at a later time?).

Appropriate corrections are required.

Any claim not specifically addressed, above, is being rejected as incorporating the deficiencies of a claim upon which it depends.

Claim Rejections - 35 USC § 103

4. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

2. Claims 1-28, 32, 33, and 35 are rejected under 35 U.S.C. 103(a) as being unpatentable over Schwabe (US 6,986,132 B1), in view of Atallah et al. (US 7,069,474 B2, hereinafter Atallah), further in view of Boonsiri et al., "Automated Component Ensemble Evaluation" (hereinafter Boonsiri).

3. As per claim 1, Schwabe teaches a method for monitoring updates to a software repository, comprising:

downloading interfaces to a stored software component, the interface having respective versions of a software interface over a network from a software repository via an application

programming interface (see Fig 11A, 11C, Fig 20C, column 17, lines 36-67, column 19, lines 19-27, column 24, lines 24-32, column 25, lines 23-36) ;

comparing the successively downloaded versions of the interfaces to detect at least one difference between them (see Figs 20C-20D, column 24, line 35-column 26, line 15); and

issuing an alert message containing an overall backward compatibility (see column 25, line 23 – column 26, line 15).

Schwabe does not explicitly teach snapshots of the software interface, and comparing the snapshots. However, a snapshot is considered as a copy of the software interface made at a particular time and applications for taking snapshots of a file are well known in the field, so it would have been obvious to one of ordinary skill in the art that a snapshot of the API definition file can be easily assembled and that comparing two snapshots of two software interfaces would be equivalent to comparing the two software interfaces directly.

Schwabe does not teach rating each detected difference according to a backward compatibility metric, determining an overall backward compatibility of the updated version based on the difference ratings, and issuing the alert to registered authors of a multi-author software design environment.

Atallah teaches a method of assessing the risk of binary compatibility failure between software products, including rating each detected difference according to a compatibility metric and determining an overall compatibility of the software products based on the difference ratings (see Figs 4A, 4B, abstract, lines 2-7, 11-14, column 5, line 64 - column 7, line 15), and issuing an alert indicating the compatibility to registered users of the risk assessment system (see Fig 3, column 2, lines 33-37, column 5, line 40 – column 6, line 3).

It would have been obvious to one of ordinary skill in the art at the time of the invention to have modified Schwabe to rate each detected difference according to a backward compatibility metric and determine an overall backward compatibility of the updated version based on the difference ratings as taught by Atallah because it is expensive and time consuming to purchase new versions of application programs and migrate the accompanying data when operating systems are updated and current versions of application programs are no longer compatible with the new version of the operating system, so the risk profile generated will allow users to better gauge whether upgrading their operating system may create binary compatibility issues with existing application software (see column 1, 42-59, and column 7, lines 5-16 of Atallah). Furthermore, it would have been obvious to one of ordinary skill in the art at the time of the invention to have modified Schwabe such that the alert is issued to registered users because it is well known in the art that registration allows personalization of the information provided and facilitates additional functionalities such as charging the user for the service.

Schwabe and Atallah do not explicitly teach a multi-author design environment and the use of a score to indicate the overall backward compatibility.

Boonsiri teaches a multi-author design environment that provides compatibility scores for different ensembles of software components (see page 40, section 1, paragraph 1, pages 41-42, section 2, page 42, section 3, last paragraph, pages 43-44, sections 3.1, 3.2, pages 50-51, section 6; EN: the K-BACEE system is considered as a multi-author design environment since system integrators access the system to determine ensembles for a design specification).

It would have been obvious to one of ordinary skill in the art at the time of the invention to have modified Schwabe and Atallah to provide a compatibility score to indicate the overall

backward compatibility as suggested by Boonsiri because the score can be used to provide a ranking (see page 51, paragraph 3 of Boonsiri). It would have been obvious to one of ordinary skill in the art at the time of the invention to have modified Schwabe and Atallah to perform the compatibility check in a multi-author environment as taught by Boonsiri because it allows system integrators to evaluate the compatibility of combining software components such that software systems can be fabricated from pre-existing components rather than develop each system from scratch (see page 40, section 1, paragraph 1, page 42, paragraphs 3-5 of Boonsiri).

4. As per claim 2, Schwabe teaches that the alert message is issued only when the overall backward compatibility indicates the updated version is not backward compatible (see column 25, line 23 – column 26, line 15).

5. As per claim 3, Schwabe does not teach that the compatibility metric comprises a table of software modifications including backward-compatible software modifications and backward-incompatible modifications.

Atallah teaches maintaining tables used during the risk assessment process to determine the compatibility of binary files (see column 6, lines 22-28, 34-39, 46-50). In determining backward compatibility between API definition files, Schwabe must have knowledge of what modifications preserve backward compatibility and what modifications break backward compatibility. It would have been obvious to one of ordinary skill in the art to maintain the backward compatible and backward incompatible modifications used in Schwabe in a database or table as taught by Atallah to facilitate easy access and maintenance.

6. As per claim 4, Schwabe further teaches that the backward incompatible software modifications include: deleting a parameter from a subroutine (see Fig 20D, column 10, lines 48-50 and column 24, lines 1-9); and deleting a field from a public data structure (see Fig 20D, column 25, lines 49-58).

7. As per claim 5, Schwabe further teaches that the backward incompatible software modifications include: adding a mandatory parameter from a subroutine (see Fig 20D, column 10, lines 48-50 and column 24, lines 1-9; EN: adding a mandatory parameter to the method changes the parameters of the method); and adding a mandatory field to a public data structure (see Fig 20D, column 25, lines 49-58).

8. As per claim 6, Schwabe further teaches that the backward-incompatible software modifications include: redefining an optional parameter as a mandatory parameter(see Fig 20D, column 10, lines 48-50 and column 24, lines 1-9); changing a parameter data type (see Fig 20D, column 26, lines 6-10); and changing a public field data type (see Fig 20D, column 26, lines 1-10).

9. As per claim 7, this claim contains limitations that are substantially similar to claim 1. Therefore, it is rejected using the same reasons as claim 1.

10. As per claim 8, Schwabe does not teach that the alert message includes a summary of each detected difference.

Atallah teaches that the alert message includes a summary of each detected difference (see Fig 4A, Fig 4B, column 6, line 31 - column 7, line 16).

11. As per claim 9, Schwabe further teaches that the detecting comprises discovering that a parameter in a version is missing from a successive version (see Fig 20D, column 10, lines 48-50 and column 26, lines 6-10; EN: changing the parameters to a method includes deleting one of the parameters).

12. As per claim 10, Schwabe further teaches that the detecting comprises discovering that a parameter is optional in a version, but the parameter is mandatory in a successive version (see Fig 20D, column 10, lines 48-50 and column 26, lines 6-10; EN: changing the parameters to a method includes changing a parameter from optional to mandatory).

13. As per claim 11, Schwabe further teaches that the detecting comprises discovering that a parameter in a version is defined as a different data type in a successive version (see Fig 20D, column 10, lines 48-50 and column 26, lines 6-10).

14. As per claims 12, Schwabe teaches the invention as claimed, including a method for monitoring updates to a software repository, comprising:

downloading snapshots of a software object where each snapshot is of a respective version (see Fig 11A, 11C, Fig 20C, column 17, lines 36-67, column 19, lines 19-27, column 24,

lines 24-32, column 25, lines 23-36; EN: the API are considered as a snapshot of the actual object) ;

comparing the successively downloaded versions of the snapshots(see Figs 20C-20D, column 24, line 35-column 26, line 15); and

detecting at least one difference between a version of an object interface and a successive version of the object interface (see Figs 20C-20D, column 24, line 35-column 26, line 15);

issuing an alert message when at least one of the detected differences indicates the successive version is not backward compatible (see column 25, line 23 – column 26, line 15).

Schwabe does not teach a multi-author design environment and issuing the alert to registered authors of the software design environment.

Boonsiri teaches a multi-author design environment that provides compatibility scores to the authors for different ensembles of software components (see page 40, section 1, paragraph 1, pages 41-42, section 2, page 42, section 3, last paragraph, pages 43-44, sections 3.1, 3.2, pages 50-51, section 6; EN: the K-BACEE system is considered as a multi-author design environment since system integrators access the system to determine ensembles for a design specification).

It would have been obvious to one of ordinary skill in the art at the time of the invention to have modified Schwabe and Atallah to provide a compatibility score to indicate the overall backward compatibility in a multi-author design environment as suggested by Boonsiri because the multi-author environment enables system integrators to evaluate the compatibility of combining software components such that software systems can be fabricated from pre-existing components rather than develop each system from scratch (see page 40, section 1, paragraph 1, page 42, paragraphs 3-5 of Boonsiri).

Schwabe and Boonsiri do not teach that the uses of the multi-author design environment are registered.

Atallah teaches a method of assessing the risk of binary compatibility failure between software products, including issuing an alert indicating the compatibility to registered users of the risk assessment system (see Fig 3, column 2, lines 33-37, column 5, line 40 – column 6, line 3).

It would have been obvious to one of ordinary skill in the art at the time of the invention to have modified Schwabe and Boonsiri to issue the alert to registered users as taught by Atallah because it is well known in the art that registration allows personalization of the information provided and facilitates additional functionalities such as charging the user for the service.

15. As per claim 13, Schwabe further teaches that the object interface comprises a list of components published by a software object residing in the object repository, said components including object properties and object methods (see Figs 11A, 18, Fig 20D, column 11, lines 51-57, and column 17, line 48 – column 18, line 5).

16. As per claim 14, Schwabe teaches the invention as claimed, including a method for monitoring updates to software interfaces, comprising:

detecting at least one difference between a first software interface of a stored software component and a second software interface of a second stored software component (see Figs 11A, 20C-20D, column 17, lines 36-67, column 24, line 35-column 26, line 15);

issuing an alert when at least one of the detected differences indicates the first software interface is not backward compatible with respect to the second software interface (see column 25, line 23 – column 26, line 15).

Schwabe does not explicitly teach taking snapshots of the software interface, and comparing the snapshots. However, a snapshot is considered as a copy of the software interface made at a particular time and applications for taking snapshots of a file are well known in the field, so it would have been obvious to one of ordinary skill in the art that a snapshot of the API definition file can be easily assembled and that comparing two snapshots of two software interfaces would be equivalent to comparing the two software interfaces directly.

Schwabe does not teach rating each detected difference according to a backward compatibility metric and issuing the alert to registered authors of a multi-author software design environment.

Atallah teaches a method of assessing the risk of binary compatibility failure between software products, including rating each detected difference according to a compatibility metric and determining an overall compatibility of the software products based on the difference ratings (see Figs 4A, 4B, abstract, lines 2-7, 11-14, column 5, line 64 - column 7, line 15), and issuing an alert indicating the compatibility to registered users of the risk assessment system (see Fig 3, column 2, lines 33-37, column 5, line 40 – column 6, line 3).

It would have been obvious to one of ordinary skill in the art at the time of the invention to have modified Schwabe to rate each detected difference according to a backward compatibility metric as taught by Atallah because it is expensive and time consuming to purchase new versions of application programs and migrate the accompanying data when operating

systems are updated and current versions of application programs are no longer compatible with the new version of the operating system, so the risk profile generated will allow users to better gauge whether upgrading their operating system may create binary compatibility issues with existing application software (see column 1, 42-59, and column 7, lines 5-16 of Atallah). Furthermore, it would have been obvious to one of ordinary skill in the art at the time of the invention to have modified Schwabe such that the alert is issued to registered users because it is well known in the art that registration allows personalization of the information provided and facilitates additional functionalities such as charging the user for the service.

Schwabe and Atallah do not explicitly teach a multi-author design environment.

Boonsiri teaches a multi-author design environment that provides compatibility scores for different ensembles of software components (see page 40, section 1, paragraph 1, pages 41-42, section 2, page 42, section 3, last paragraph, pages 43-44, sections 3.1, 3.2, pages 50-51, section 6; EN: the K-BACEE system is considered as a multi-author design environment since system integrators access the system to determine ensembles for a design specification).

It would have been obvious to one of ordinary skill in the art at the time of the invention to have modified Schwabe and Atallah to provide a compatibility score to indicate the overall backward compatibility as suggested by Boonsiri because the score can be used to provide a ranking (see page 51, paragraph 3 of Boonsiri). It would have been obvious to one of ordinary skill in the art at the time of the invention to have modified Schwabe and Atallah to perform the compatibility check in a multi-author environment as taught by Boonsiri because it allows system integrators to evaluate the compatibility of combining software components such that

software systems can be fabricated from pre-existing components rather than develop each system from scratch (see page 40, section 1, paragraph 1, page 42, paragraphs 3-5 of Boonsiri).

17. As per claim 15, Schwabe does not teach rating each detected difference according to a predetermined difference metric; determining an overall difference between the first software interface and the second software interface based on the difference ratings; and incorporating the overall difference into the alert message.

Atallah teaches a method of assessing the risk of binary compatibility failure between software products, including rating each detected difference according to a compatibility metric and determining an overall compatibility of the software products based on the difference ratings (see Figs 4A, 4B, abstract, lines 2-7 and lines 11-14, column 5, line 64 - column 7, line 15).

It would have been obvious to one of ordinary skill in the art at the time of the invention to have modified Schwabe to rate each detected difference according to a predetermined metric and determine an overall difference between the first interface and the second interface based on the difference ratings as taught by Atallah because it is expensive and time consuming to purchase new versions of application programs and migrate the accompanying data when operating systems are updated and current versions of application programs are no longer compatible with the new version of the operating system, so the risk profile generated will allow users to better gauge whether upgrading their operating system may create binary compatibility issues with existing application software (see column 1, 42-59, and column 7, lines 5-16 of Atallah).

18. As per claim 16, Schwabe does not teach assigning a user to the first snapshot; and issuing the alert message to the user.

Atallah teaches a method of assessing the risk of binary compatibility failure between software products, including assigning a user to the application being assessed (see column 5, lines 40-63) and issuing the risk assessment report to the user (see column 7, lines 5-16).

It would have been obvious to one of ordinary skill in the art at the time of the invention to have modified Schwabe to assign a user to the first snapshot; and issuing the alert message to the user as taught by Atallah because it is expensive and time consuming to purchase new versions of application programs and migrate the accompanying data when operating systems are updated and current versions of application programs are no longer compatible with the new version of the operating system, so the risk profile generated will allow users to better gauge whether upgrading their operating system may create binary compatibility issues with existing application software (see column 1, 42-59, and column 7, lines 5-16 of Atallah).

19. As per claim 17, Schwabe does not teach incorporating a summary of each detected difference into the alert message.

Atallah teaches a method of assessing the risk of binary compatibility failure between software products, including incorporating a summary of each detected compatibility difference into a report (see Fig 4A, Fig 4B, column 6, line 31 - column 7, line 16).

It would have been obvious to one of ordinary skill in the art at the time of the invention to have modified Schwabe to incorporate a summary of each detected difference into the alert message as taught by Atallah because it is expensive and time consuming to purchase new

versions of application programs and migrate the accompanying data when operating systems are updated and current versions of application programs are no longer compatible with the new version of the operating system, so the risk profile generated will allow user to better gauge whether upgrading their operating system may create binary compatibility issues with existing application software (see column 1, 42-59, and column 7, lines 5-16 of Atallah).

20. As per claim 18, Schwabe further teaches that the API definition file has at least one record, said record including at least one attribute (see Fig 20D, column 25, line 37 – column 26, line 10; En: attributes of classes, interfaces, fields and methods, are all considered as separate records in the API definition file) While Schwabe does not teach organizing the records in a table, it would have been obvious that a table structure can be used since a table is a well known data structure in the art. Furthermore, it would have been obvious that a snapshot of the API definition file would have the records because the snapshot is considered as just a copy of the API definition file.

21. As per claim 19, Schwabe further teaches that the record describes an object property (see Fig 20D, column 25, lines 49-58; EN: the fields of a class are considered as properties of the class since the fields of a class describe the class and class properties are considered object properties because it is well known in the art that an object is just an instance of a class).

22. As per claim 20, Schwabe further teaches that the attribute indicates a data type of the object property (see Fig 20D, column 25, lines 49-58, column 26, lines 1-5; EN: the data type of the fields are considered data type of the object property).

23. As per claim 21, Schwabe further teaches that the attribute indicates a data length of the object property (see Fig 20D, column 25, lines 49-58, column 26, lines 1-5; EN: length information is implied by the type information since a type of int requires a different amount of storage than a type of float).

24. As per claim 22, Schwabe further teaches that the record describes an object method (see Fig 20D, item 1655, and column 26, lines 6-10).

25. As per claim 23, Schwabe further teaches that the records describe an object method parameter (see Fig 20D, item 1655, and column 26, lines 6-10; EN: the object method parameter is described in the method signature).

26. As per claim 24, Schwabe further teaches that the attribute indicates a data type of the object method parameter (see Fig 20D, item 1655, and column 26, lines 6-10; EN: the data type of the object method parameter is described in the method signature).

27. As per claim 25, this is a computer claim containing limitations that are substantially similar to claim 1. Therefore, it is rejected using the same reasons as claim 1.

28. As per claim 26, Schwab does not teach that the computer has the means to rate each detected difference according to a backward compatibility metric.

Atallah teaches a method of assessing the risk of binary compatibility failure between software products, including rating each detected difference according to a compatibility metric and determining an overall compatibility of the software products based on the difference ratings (see Figs 4A, 4B, abstract, lines 2-7 and lines 11-14, column 5, line 64 - column 7, line 15).

It would have been obvious to one of ordinary skill in the art at the time of the invention to have modified Schwab to rate each detected difference according to a backward compatibility metric as taught by Atallah because it is expensive and time consuming to purchase new versions of application programs and migrate the accompanying data when operating systems are updated and current versions of application programs are no longer compatible with the new version of the operating system, so the risk profile generated will allow user to better gauge whether upgrading their operating system may create binary compatibility issues with existing application software (see column 1, 42-59, and column 7, lines 5-16 of Atallah).

29. As per claim 27, Schwab does not teach that the computer has means to incorporate the ratings and the detected differences into the alert message.

Atallah teaches generating a report to indicate the binary compatibility risk profile with records to indicate failure, high risk, low risk, and guaranteed eligible (see Figs 4A, 4B, abstract, lines 10-16, column 5, line 64 - column 7, line 15).

30. As per claim 28, this is a computer-readable medium claim containing limitations that are substantially similar to claim 1. Therefore, it is rejected using the same reasons as claim 1.

31. As per claim 32, this is a system claim containing limitations that are substantially similar to claim 1. Therefore, it is rejected using the same reasons as claim 1.

32. As per claim 33, Schwabe the API definition file comprises: the relevant data objects, subroutines, and associated attributes; where the proceeding are compiled for a selection of interfaces as well as any interfaces that are associated with the selected interfaces (see Fig 20D, steps 1635- 1655, column 25, line 37 - column 26, line 10; EN: the selected interfaces are the implemented interfaces and the interfaces associated with the selected interfaces are the superinterfaces). While Schwabe does not explicitly teach a snapshot of the API definition file, it would have been obvious that a snapshot of the API definition file would have the same information as the API definition file because the snapshot is considered as just a copy of the API definition file.

33. As per claim 35, Schwabe does not teach wherein the alert message is issued if no difference is detected between the two snapshots being compared.

Atallah teaches issuing an alert message if no difference is detected between the two snapshots being compared (see column 6, line 64 – column 7, line 16).

It would have been obvious to one of ordinary skill in the art at the time of the invention to have modified Schwabe to issue an alert message if no difference is detected between the two

snapshots being compared as taught by Atallah because the report enables the user to better gauge binary compatibility issues (see column 7, lines 2-15 of Atallah).

34. Claims 29-31 are rejected under 35 U.S.C. 103(a) as being unpatentable over Schwabe (US 6,986,132 B1) in view of Atallah et al. (US 7,069,474 B2, hereinafter Atallah), further in view of Boonsiri et al., "Automated Component Ensemble Evaluation" (hereinafter Boonsiri), as applied to claim 28 above, further in view of Theodossy et al. (US 6,898,768 B1, hereinafter Theodossy).

35. As per claim 29, Schwabe, Atallah, and Boonsiri do not teach that the plurality of instructions stored on the machine-readable medium is executed in response to an external trigger.

Theodossy teaches a method for compatibility receiving a compatibility triggering event and verifying compatibility in response to the triggering event (see Fig 5, abstract, lines 4-7, column 3, lines 32-35, and column 6, lines 23-67).

It would have been obvious to one of ordinary skill in the art at the time of the invention to have modified Schwabe, Atallah, and Boonsiri such that the instructions are executed in response to an external trigger as taught by Theodossy because compatibility is critical for all upgrades and it is important to verify that compatibility exists for hardware revisions and software revisions prior to experiencing problems relating to incompatibility, and the triggering event allows this verification to occur automatically (see column 2, lines 36-45 and column 3, lines 35-40 of Theodossy).

36. As per claim 30, Schwabe, Atallah, and Boonsiri do not teach that the external trigger comprises a received notification of an update to the second software interface.

Theodossy teaches that the triggering event can be an attempt to perform a software upgrade on a system component (see column Fig 4, column 2, lines 48-55, column 5, line 52 – column 6, line 13, and column 6, line 23-63). While Theodossy does not specifically teach as such, it would have been obvious to one of ordinary skill in the art that the triggering event can be a notification of an update to the second software interface since an triggering event can be any event that necessitates the system to perform a compatibility verification to ensure optimal performance (see column 6, lines 29-33 of Theodossy) and update to a software interface would have an impact on other software interfaces relying on the updated software interface.

37. As per claim 31, Schwabe, Atallah, and Boonsiri do not teach that the external trigger comprises a received notification of a scheduled event.

Theodossy teaches receiving a compatibility triggering event (see column Fig 4, column 2, lines 48-55, column 5, line 52 – column 6, line 13, and column 6, line 23-63). While Theodossy does not explicitly teach that the triggering event is a scheduled event, it would have been obvious to one of ordinary skill in the art that the triggering event could be scheduled since the triggering event can be upgrades on a system component and system upgrades can be a complex process requiring much planning (see column 1, lines 12-13).

38. Claim 34 is rejected under 35 U.S.C. 103(a) as being unpatentable over Schwabe (US 6,986,132 B1) in view of Atallah et al. (US 7,069,474 B2, hereinafter Atallah), further in view of

Boonsiri et al., "Automated Component Ensemble Evaluation" (hereinafter Boonsiri), as applied to claim 1 above, further in view of Kiessig et al., (US 7,289,973 B2, hereinafter Kiessig).

39. As per claim 34, Schwabe, Atallah, and Boonsiri do not teach providing an option to store snapshots in a snapshot history.

Kiessig teaches a method of storing snapshots in a snapshot history (see column 5, lines 13-22, 27-36, 44-57).

It would have been obvious to one of ordinary skill in the art at the time of the invention to have modified Schwabe, Atallah, and Boonsiri to store snapshots in a snapshot history as taught by Kieesig because it allows users to restore past snapshot versions as the current snapshot (see column 5, lines 13-22 of Kiessig).

Response to Arguments

40. Rejections of claims under §103(a):

41. As per independent claim 1, Applicant argued that Atallah does not teach "rating each detected difference according to a backward compatibility metric" because in Atallah's system, only one rating is given, and the rating is only an overall rating for the binary. Applicant's arguments have been fully considered and Examiner respectfully disagrees. Examiner submits that Atallah rates each difference found for the binary differently because Atallah teaches generating a record for each compatibility test (see Fig 4A, Fig 4B, column 6, line 31 - column 7, line 4). The result of each compatibility test that resulted in a yes answer is considered as one

difference (i.e., binary invoked unsupported symbols, binary invokes problem libraries, binary invokes changed libraries, etc.), and each of these differences are rated separately and differently (i.e., a failure record is generated for binary invoking unsupported symbols and a high risk record is generated for binary invoking changed libraries) because a separate record is generated for each difference regardless of answers to the previous compatibility tests as indicated by flow graph on Fig 4A and Fig 4B where flow continues to the next compatibility test after the generation of a record. Furthermore, Atallah teaches generating an overall compatibility rating based on the difference rating because Atallah teaches "Rather than presenting a simple pass or fail output, the risk assessment system assigns a risk level to the application based on the outcome of the test administered by the appcert application" (column 5, line 67 – column 6, line 3).

42. As per claim 1, Applicant also argued that neither Schwabe nor Atallah teaches the limitation of "issuing an alert message to registered authors of the software design environment". Examiner submits that the combination of Atallah and newly cited prior art Boonsiri teaches this limitation as per the reasons given above in the rejection of claim 1.

43. As per claim 8, Applicant argued that Atallah does not teach "alert messages [which] include[] a summary of each detected difference". Applicant's arguments have been fully considered and Examiner respectfully disagrees. Examiner submits that Atallah generates a record for each difference found for the binary because Atallah teaches generating a record for each compatibility test (see Fig 4A, Fig 4B, column 6, line 31 - column 7, line 4). The result of

each compatibility test that resulted in a yes answer is considered as one difference (i.e., binary invoked unsupported symbols, binary invokes problem libraries, binary invokes changed libraries, etc.), and each of these differences are recorded separately because a separate record is generated for each difference regardless of answers to the previous compatibility tests as indicated by flow graph on Fig 4A and Fig 4B where flow continues to the next compatibility test after the generation of a record. Atallah further teaches that the generated records are formatted into a report to be transmitted to the end user (see column 7, lines 5-16), so the report which is considered the alert message contains a summary of each detected difference as claimed.

44. As per claim 16, Applicant argued that Atallah does not teach "assigning a user to [a] snapshot" and "issuing the alert message to the user". Applicant's arguments have been fully considered and Examiner respectfully disagrees. Atallah teaches the user providing registration information to the risk assessment system and then transmitting binary files to the risk assessment system (see column 5, lines 40-63), where the risk assessment system performs binary analysis on the received files and produces a report about the binary compatibility and transmits the report back to the user (see column 5, line 64 – column 7, line 16). Examiner submits that the act of transmitting the binary file by the registered user to the risk assessment system assigns the binary file to the registered user because the risk assessment system must associate the binary file with the registered users in order to provide the binary compatibility report generated to the registered user.

Conclusion

45. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

- Spinard et al. (US 6,873,935 B2) is cited to teach a system and method for statically checking source code including identifying changes to identify potential incompatibilities between two software releases.
- Seacord et al., "K-BACEE: knowledge-based automated component ensemble evaluation", 2001, Proceedings of the 27th Euromicro Conference.
  This document is cited to teach generating a compatibility score for different component ensembles.

46. Applicants' amendment necessitated the new ground(s) of rejection presented in this office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP §706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

47. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Jue S. Wang whose telephone number is (571) 270-1655. The examiner can normally be reached on M-Th 7:30 am - 5:00pm (EST).

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Lewis Bullock can be reached on 571-272-3759. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Lewis A. Bullock, Jr./
Supervisory Patent Examiner, Art Unit 2193

Jue Wang
Examiner
Art Unit 2193